

(2) Japanese Patent Application Laid-Open No. JP8-115206(1996)

“Floating Point Number Operation Device”

The following is an extract relevant to the present application.

5

A floating point number operation device including a means for adding three selecting circuits, three shifters, a complement circuit, and two data buses and providing an add/subtract circuit with an output of a multiply/divide normalization circuit by the first two bits thereof, a means for complementing an adder/subtractor adder circuit input data, and a
10 means for bit shifting the adder/subtractor adder circuit input data, an adder/subtractor adder circuit result and an adder circuit overflow to be returned to the adder/subtractor adder circuit input again.

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-115206

(43) 公開日 平成8年(1996)5月7日

(51) Int.Cl. ⁶	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 7/552	B			
7/00				
7/52	3 1 0 C	8323-5E	G 0 6 F 7/ 00	1 0 1 W
審査請求 未請求 請求項の数 1 O L (全 14 頁)				

(21) 出願番号 特願平6-249089

(22) 出願日 平成6年(1994)10月14日

(71) 出願人 000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目6番地

(71) 出願人 000233011
日立コンピュータエンジニアリング株式会社
神奈川県秦野市堀山下1番地

(72) 発明者 瀧口 誠
神奈川県秦野市堀山下1番地 日立コンピュータエンジニアリング株式会社内

(74) 代理人 弁理士 小川 勝男

最終頁に続く

(54) 【発明の名称】 浮動小数点演算装置

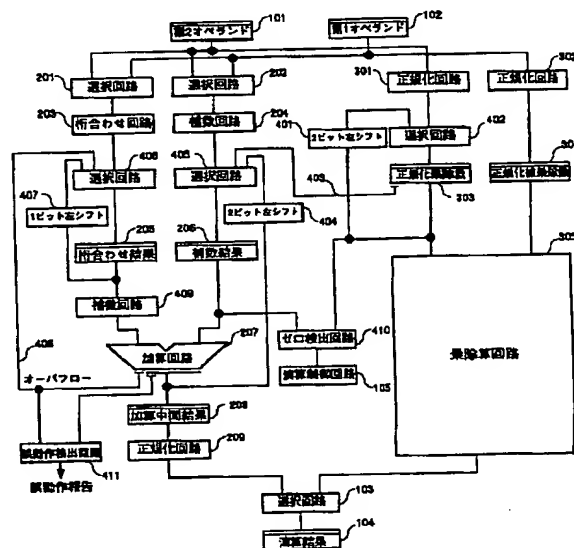
(57) 【要約】

【目的】 一般的な構成の浮動小数点演算装置に少量の回路を付加することにより平方根演算器が可能となる浮動小数点演算装置を提供する。

【構成】 3個の選択回路、3個のシフタ、補数回路、2本のデータバスを付加し、乗除算器正規化回路の出力を先頭から2ビットずつ加減算回路に供給する手段、加減算器加算回路入力データを補数化する手段、加減算器加算回路入力データと加減算器加算回路結果と加算回路オーバーフローをビットシフトし再び加減算器加算回路入力に戻す手段を設けた浮動小数点演算装置。

【効果】 一般的な構成の浮動小数点演算装置に少量の回路を付加することにより、非回復型平方根アルゴリズムによる平方根演算可能となる。

図 1



【特許請求の範囲】

【請求項 1】 加減算器と乗除算器とで構成される浮動小数点演算装置において、前記乗除算器の正規化乗除数レジスタの先頭 2 ビットを前記加減算器に転送する第 1 データバスと、前記加減算器に転送した 2 ビットのデータをシフトアウトして正規化乗除数レジスタにセットするための第 1 の 2 ビット左シフト及び選択回路と、前記加減算器の加算回路の出力結果を 2 ビット左シフトする第 2 の 2 ビット左シフトと、加減算器の補数回路の出力、第 2 の 2 ビット左シフトの出力、及び乗除算器からの正規化乗除数レジスタの先頭 2 ビットをマージしたデータを選択する加減算器の補数結果レジスタ入力選択回路と、加減算器の加算回路の結果の最上位ビットを加減算器の桁合わせ結果レジスタ入力選択回路に転送する第 2 データバスと、加減算器の桁合わせ結果レジスタを 1 ビット左シフトする第 3 の 1 ビット左シフトと、加減算器の桁合わせ回路の出力、第 3 の 1 ビット左シフトの出力、及び加算回路の結果の最上位ビットをマージしたデータを選択する加減算器の桁合わせ結果レジスタ入力選択回路と、加減算器の桁合わせ結果レジスタの出力を補数化する桁合わせ結果のレジスタ出力補数回路と、加減算器の補数結果レジスタと乗除算器の正規化乗除数レジスタがゼロであるか否かを検出し、演算制御回路にゼロ情報を送出するゼロ検出回路とを備えたことを特徴とした浮動小数点演算装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、浮動小数点演算装置に関し、特に、既存の演算器回路を最大限に使用し、最小限の回路を付加することにより平方根演算を可能にする浮動小数点演算装置に関するものである。

【0002】

【従来の技術】従来、平方根の計算をハードウェアで実現する場合の手法として非回復型平方根アルゴリズムが知られている。これは、図 9 に示すように、加減算とシフトにより 1 回の開平ステップで 1 ビットの部分平方根を求める手法である。この手法による平方根演算については、例えば「コンピュータの高速演算方法」（堀越彌訳、近代科学社、昭和 55 年 9 月 1 日発行）第 357 頁から 359 頁に論じられている。

【0003】非回復型平方根アルゴリズムと冗長 2 進表現による平方根専用演算器を用い、1 回の開平ステップで 2 ビットの部分平方根を求める発明が特開平 3-296132 公報に開示されている。

【0004】非回復型平方根アルゴリズムと類似している手法に回復型平方根アルゴリズムがある。これは、図 10 に示すように、減算とシフトにより平方根を求める手法である。この手法の平方根演算に関するものには、例えば特開平 1-155435 公報が挙げられる。特開平 1-155435 公報では回復型平方根アルゴリズム

を除算器上で実現した平方根演算器の発明について開示されている。これは回復型平方根アルゴリズムと高基数非回復型除算アルゴリズムの手法が類似しているため除算器に必要な回路を付加して平方根演算を実行可能にしたものである。前記図 9 及び図 10 において、2 は加減算器、3 は乗除算器、101 は第 2 オペランドレジスタ、102 は第 1 オペランドレジスタ、103、201、202 は選択回路、104 は演算結果レジスタ、203 は桁合わせ回路、204 は補数回路、205 は桁合わせ結果レジスタ、206 は補数結果レジスタ、207 は加算回路、208 は加算中間結果レジスタ、209、301、302 は正規化回路、303 は正規化乗除数レジスタ、304 は正規化被乗除数レジスタ、305 は乗除算回路である。

【0005】

【発明が解決しようとする課題】従来技術の非回復型平方根アルゴリズムと冗長 2 進表現による発明は、1 回の開平ステップで 2 ビットの部分平方根を求めることが可能であるため性能的には優れているが専用演算器を設けるためにハードウェア物量が増加してしまう。

【0006】科学技術計算の分野で用いられる平方根演算命令は、命令処理の高速化も要求されるが、命令の使用頻度が比較的少ないため専用演算器を設けることによる性能向上の効果よりハードウェア物量増加の悪影響の方が大きい場合がある。

【0007】従来技術の除算器上で平方根演算を実行する発明は、既存の演算器に平方根演算で必要な回路を付加するためハードウェア物量の増加は少量ですむが、高基数非回復型除算アルゴリズムと異なる除算アルゴリズムの除算器では実現不可能な場合がある。例えば、乗算器を用いた収束型除算器では実現不可能である。

【0008】本発明の目的は、一般的な構成の浮動小数点演算装置に少量の回路を付加することにより、平方根演算が可能となる浮動小数点演算装置を提供することにある。

【0009】本発明の前記ならびにその他の目的及び新規な特徴は、本明細書の記載及び添付図面によって明らかにするであろう。

【0010】

【課題を解決するための手段】本願において開示された発明のうち代表的なものの概要を簡単に説明すれば、以下のとおりである。

【0011】加減算器と乗除算器とで構成される浮動小数点演算装置において、前記乗除算器の正規化乗除数レジスタの先頭 2 ビットを前記加減算器に転送する第 1 データバスと、前記加減算器に転送した 2 ビットのデータをシフトアウトして正規化乗除数レジスタにセットするための第 1 の 2 ビット左シフト及び選択回路と、前記加減算器の加算回路の出力結果を 2 ビット左シフトする第 2 の 2 ビット左シフトと、加減算器の補数回路の出力、

第2の2ビット左シフトの出力、及び乗除算器からの正規化乗除数レジスタの先頭2ビットをマージしたデータを選択する加減算器の補数結果レジスタ入力選択回路と、加減算器の加算回路の結果の最上位ビットを加減算器の桁合わせ結果レジスタ入力選択回路に転送する第2データバスと、加減算器の桁合わせ結果レジスタを1ビット左シフトする第3の1ビット左シフトと、加減算器の桁合わせ回路の出力、第3の1ビット左シフトの出力、及び加算回路の結果の最上位ビットをマージしたデータを選択する加減算器の桁合わせ結果レジスタ入力選択回路と、加減算器の桁合わせ結果レジスタの出力を補数化する桁合わせ結果のレジスタ出力補数回路と、加減算器の補数結果レジスタと乗除算器の正規化乗除数レジスタがゼロであるか否かを検出し、演算制御回路にゼロ情報を送出するゼロ検出回路とを備えたものである。

【0012】また、信頼性を確保した遅延時間の短縮は、前記の加減算器の加算回路の結果最上位ビットのかわりに、加減算器の加算回路の結果のオーバーフローを加減算器の桁合わせ結果レジスタ入力選択回路の転送データバスとして用いること、及び加減算器の加算回路の結果最上位ビットの値と加算回路の結果オーバーフローの値を比較する誤動作検出回路を設けたものである。

【0013】

【作用】前述の手段によれば、乗除算器の正規化乗除数レジスタの先頭2ビットを加減算器に転送し、この加減算器に転送した2ビットのデータを2ビット左シフトと選択回路でシフトアウトして正規化乗除数レジスタにセットし、前記加減算器の加算回路の結果を2ビット左シフトし、加減算器の補数回路の出力、加減算器の加算回路の結果の2ビット左シフト出力、乗除算器からの正規化乗除数レジスタの先頭2ビットをマージしたデータを選択し、加減算器の加算回路の結果の最上位ビットを加減算器の桁合わせ結果レジスタ入力選択回路に転送し、加減算器の桁合わせ結果レジスタを1ビット左シフトし、加減算器の桁合わせ結果レジスタ入力選択回路で加減算器の桁合わせ回路出力、加減算器の桁合わせ結果レジスタの1ビット左シフトは出力、及び加算回路の結果の最上位ビットをマージした（重ねた）データを選択し、桁合わせ結果レジスタ出力補数回路で加減算器の桁合わせ結果レジスタ出力を補数化し、ゼロ検出回路で加減算器の補数結果レジスタと乗除算器の正規化乗除数レジスタがゼロであるか否かを検出し、演算制御回路にゼロ情報を送出することにより、加減算器と乗除算器から構成される一般的な浮動小数点演算装置に、3個のシフト、補数回路、2個のデータバスという少量の回路を付加するだけで、平方根演算可能な浮動小数点演算装置を実現することができる。

【0014】また、さらに加算回路の結果の最上位ビットとオーバーフロー値を比較する誤動作検出回路を設けることにより加算回路のオーバーフローの誤動作を検出する

ことができる。

【0015】

【実施例】以下、本発明による実施例を図面を用いて詳細に説明する。

【0016】図1は、本発明による一実施例の浮動小数点演算装置の仮数部の概略構成を示すブロック図である。図1において、101は第2オペランドレジスタ、102は第1オペランドレジスタ、103は加減算器と乗除算器の演算結果を選択する選択回路、104は演算結果を格納する演算結果レジスタ、105は演算器全体の制御を行なう演算制御回路である。

【0017】201～209は、ロード命令や加減算命令を実行する加減算器の構成要素であり、201は指数小側の仮数データを選択する選択回路、202は指数大側の仮数データを選択する選択回路、203は指数小側仮数データの桁合わせを行なうシフト、204は指数大側仮数データを真の減算時に補数化する補数回路、205は桁合わせ回路出力を保持する桁合わせ結果レジスタ、206は補数回路出力を保持する補数結果レジスタ、207は桁合わせデータと補数回路データを加算する加算回路、208は加算回路の結果を保持する加算中間結果レジスタ、209は加算中間結果データを正規化する正規化回路である。

【0018】301～305は、乗除算命令を実行する乗除算器の構成要素であり、301は第2オペランドを正規化する正規化回路、302は第1オペランドを正規化する正規化回路、303は正規化回路301の出力を保持する正規化乗除数レジスタ、304は正規化回路302の出力を保持する正規化被乗除数レジスタ、305は正規化乗除数と正規化被乗除数から乗除算結果を求める乗除算回路である。401～411は、本発明により浮動小数点演算装置に付加した平方根演算のための平方根演算回路であり、401は正規化乗除数レジスタの値を2ビット左シフトする第1の2ビット左シフト、402は正規化回路301出力、第1の2ビット左シフト401出力を選択する正規化乗除数レジスタ入力選択回路である。

【0019】403は正規化乗除数レジスタ303に格納された a_{2k+1} と a_{2k+2} を選択回路405に転送するデータバスである。404は加算回路207の出力 R_k を2ビット左シフトする第2の2ビット左シフトである。405は補数回路204出力、データバス403、第2の2ビット左シフト404の出力を選択する補数結果レジスタ入力選択回路である。

【0020】406は加算回路207のオーバーフロービットを選択回路408に転送するデータバスである。407は桁合わせ結果レジスタ205に格納した $q_1 \sim q_k - 1$ を1ビット左シフトする第3の1ビット左シフトである。408は桁合わせ回路203出力、加算回路207のオーバーフローデータバス406、第3の1ビット左シ

フタ407の出力を選択する桁合わせ結果レジスタ入力選択回路である。

【0021】409は桁合わせ結果レジスタ205の出力を q_k の値により補数化する桁合わせ結果レジスタ補数回路である。410は補数結果レジスタ206、正規化乗除数レジスタ303に格納されている R_k と $a_{2k+1} \sim a_n$ がゼロであることを検出するゼロ検出回路である。ゼロ検出回路410で検出されたゼロ情報は、演算制御回路105に転送され繰返し演算の終了条件に使われる。

【0022】411は平方根演算の繰返し演算時に加算回路207の結果の最上位ビットの値とオーバフローの値を比較する誤動作検出回路である。加算回路207の結果の最上位ビットとオーバフロー値が排他的でないとき誤動作報告を行なう。

【0023】一般的な浮動小数点演算装置の加算器は、仮数部7バイトにガードデジット0.5バイトを合わせた7.5バイト（（1, 0）～（8, 3））の加減算を可能とするため7.5バイト幅のデータを処理できる。

【0024】図2は、本実施例の前記正規化乗除数レジスタ入力選択回路402の動作を説明するための図である。この正規化乗除数レジスタ入力選択回路402は、図2に示すように、平方根命令以外（乗除算命令）と平方根命令の初期設定時は正規化回路の出力を選択し、平方根演算の繰返し演算時は、2ビット左シフト401の値を選択する。2ビット左シフト401と選択回路402により平方根演算時、正規化乗除数レジスタ303には $a_{2k+1} \sim a_n$ の値を左詰めで格納できる。そのため選択回路405には a_{2k+1} と a_{2k+2} 、ゼロ検出回路410には $a_{2k+3} \sim a_n$ を容易に供給できる。

【0025】図3は、本実施例の前記補数結果レジスタ入力選択回路405の動作を説明するための図である。この補数結果レジスタ入力選択回路405は、図3に示すように、平方根命令以外（加減算、ロード命令）時は正規化回路の出力を選択し、平方根演算の初期設定時はデータバス403の a_1 、 a_2 を選択する。平方根演算の繰返し演算時には、上位ビットに2ビット左シフト404出力、下位ビットにはデータバス403の a_{2k+1} 、 a_{2k+2} を選択する。平方根演算で $k > n+1$ が成立した場合の終了処理では丸め処理を行なうために、 q_{57} と同じビット位置に1を立てるため0……0100を選択する。平方根演算の $R_k = 0$ 、かつ $a_{2k+1} \sim a_n = 0$ が成立した途中終了の場合の終了処理時はゼロを選択する。

【0026】図4は、前記桁合わせ結果レジスタ入力選択回路408の動作を説明するための図である。この桁合わせ結果レジスタ入力選択回路408は、図4に示すように、平方根命令以外（加減算、ロード命令）時は桁合わせ回路の出力を選択し、平方根演算の初期設定時はゼロを選択する。平方根演算の繰返し演算時、終了処理時は上位ビットに1ビット左シフト407の出力の $q_1 \sim$

q_{k-1} を選択し、次のビットにはオーバフローバス406の q_k を選択し、最下位の2ビットには0を選択する。

【0027】図5は、前記桁合わせ結果レジスタ補数回路409の動作を説明するための図である。この桁合わせ結果レジスタ補数回路は平方根命令以外（加減算、ロード命令）時、平方根演算の終了時は桁合わせ結果レジスタ出力を選択する。平方根演算の繰返し演算で $q_k = 1$ 時は桁合わせ結果レジスタの出力をビットごとに反転し、最下位の2ビットを1としたデータを選択する。平方根演算の繰返し演算で $q_k = 0$ 時は桁合わせ結果レジスタの出力の最下位の2ビットを1としたデータを選択する。

【0028】図6は、本実施例の平方根演算器で実行する平方根命令の処理手順を示すフローチャートであり、長精度浮動小数点平方根命令の処理フローであるが、短精度浮動小数点平方根命令の場合には被開平方数と演算結果の仮数が56ビットから24ビットに変わるだけである。

【0029】本実施例で演算する平方根命令は、図6に示すように、最初に、被開平方数が正、ゼロ、負であるか調べる（ステップ601）。負の場合には平方根例外の割込みを報告する（ステップ606、607）。ゼロの場合には演算処理を行わず演算結果をゼロにして終了する（ステップ608）。正の場合には被開平方数を指数が偶数になる様に正規化する（ステップ602）。

【0030】これは次に行なう指数の演算処理（ステップ603）、すなわち、正規化した指数の値を2で割り演算結果指数を求める処理（ステップ611）において割り切れるよう、あらかじめ指数を偶数とする必要があることによる。仮数の演算処理（ステップ604）は仮数平方根を求める処理（ステップ612）と丸め処理（ステップ613）から構成される。

【0031】仮数平方根を求める処理（ステップ612）は非回復型平方根アルゴリズムで57ビットの平方根を求めることにより実現される。57ビットの平方根を求めるのは、その後に行なう丸め処理（ステップ613）のためで、演算結果より1ビット余分に平方根を求める。

【0032】丸め処理（ステップ613）では、余分に求めた最終ビットに1を加える。最後に、指数演算処理と仮数演算処理の演算結果を結合した値が演算結果となり（ステップ605）、平方根命令演算を終了する。

【0033】図7は、本実施例の浮動小数点演算装置の動作を模式的に示した図である。図7では、選択回路で選択したデータがわかるように記述したため選択回路自身の記述を省略した。

【0034】平方根演算では、桁合わせ結果レジスタ205は、 $q_1 \sim q_{k00}$ を格納するために使用し、補数結果レジスタ206は R_k を格納するために使用する。ま

た、補数回路409の出力が $(\pm Dk)$ となる。

【0035】初期設定動作を説明する。図7に示すように、最初に、第2オペランドレジスタ101に被開平方数がセットされる。次に、正規化回路301で指数が偶数になるように、被開平方数仮数部を正規化し、そのデータを正規化乗除数レジスタ303にセットする。次に、桁合わせ結果レジスタ205には0をセットし、補数結果レジスタ206には上位ビットを0とし最下位の2ビットには正規化乗除数レジスタ303の最上位の2ビット(a_1, a_2)をセットする。

【0036】正規化乗除数レジスタ303は、 a_1, a_2 を加算器に転送すると、2ビット左シフト402により a_1, a_2 をシフトアウトしデータ $a_3 \sim a_n$ を格納する。正規化乗除数レジスタ303の値は R_0 となる。ここま

でが初期設定動作である。

【0037】次に、繰返し演算動作を説明する。桁合わせ結果レジスタ205にセットされている $q_1 \sim q_k$ は q_k の値に従い補数回路409で補数化し $(\pm Dk)$ とする。補数回路409での補数化は、先に説明した図5に示すように行なう。補数回路409の出力 $(\pm Dk)$ は、補数結果レジスタ206にセットされている R_k と加算回路207で加算する。

【0038】加算回路207の結果は R_{k+1} であり、加算回路207のオーバーフロー406は q_{k+1} である。加算回路207の結果 R_{k+1} は、2ビット左シフト404でシフトされ、正規化乗除数レジスタ303の最上位2ビット a_{2k+1}, a_{2k+2} を最下位ビットに付加して補数結果レジスタ206にセットする。

【0039】正規化乗除数レジスタ303は、 a_{2k+1}, a_{2k+2} を加算器に転送すると2ビット左シフト402により、 a_{2k+1}, a_{2k+2} をシフトアウトしデータ $a_{2k+3} \sim a_n$ を格納する。加算回路207のオーバーフロー206である q_{k+1} は、桁合わせ結果レジスタ205を1ビット左シフト407でシフトした $q_1 \sim q_{k-1}$ の下位ビットに付加して桁合わせレジスタ205にセットする。

【0040】誤動作検出回路411は、加算回路207の結果の最上位ビットとオーバーフロー値を比較し、これらが排他的でないとき誤動作報告を行なう。ここま

でが繰返し演算の1回分である。繰返し演算は1マシンサイクルで1回演算し、繰返し演算1回では1ビットの q_k が求まる。繰返し演算を n 回繰返し q_1 から q_n を求める。本実施例の浮動小数点演算装置で演算する長精度平方根命令では57回繰返し q_1 から q_{57} を求める。

【0041】次に、終了処理動作を説明する。終了動作は $k > n+1$ (図7の非回復型平方根アルゴリズムでは $k > n$ までだが、本実施例の浮動小数点演算装置で演算する平方根命令では1ビット余分に q_k を求め、丸め処理を行なうため $k > n+1$ とする)が成立した場合、及び $R_k = 0$ 、かつ $a_{2k+1} \sim a_n = 0$ が成立した途中終了の場合がある。

【0042】 $k > n+1$ が成立した場合には、 q_{n+1} のビットに1を加える丸め処理を行なう。丸め処理では、最初に補数結果レジスタ206に0……0100を選択してセットする。

【0043】次に、補数回路409は、桁合わせレジスタに格納されている $q_1 \sim q_{n+1}$ を素通りさせ、その出力と補数回路206の0……0100を加算回路207で加算する。207の加算結果は、加算中間結果レジスタ208にセットする。次に、加算中間結果レジスタ208にセットした値を正規化回路209で q_1 が最も左のビットに来るようにシフトし、 q_n より右のビットを切捨て演算結果レジスタにセットし演算を終了する。

【0044】 $R_k = 0$ 、かつ $a_{2k+1} \sim a_n = 0$ が成立した途中終了の場合は、最初に、補数結果レジスタ206にゼロを選択してセットする。次に、補数回路409は桁合わせレジスタに格納されている $q_1 \sim q_k$ を素通りさせ、その出力と補数回路206のゼロを加算回路207で加算する。加算回路207の加算結果は、加算中間結果レジスタ208にセットする。次に、加算中間結果レジスタ208にセットした値を正規化回路209で q_1 が最も左のビットに来るようにシフトし演算結果レジスタ104にセットし演算を終了する。

【0045】以上の説明からわかるように、本実施例によれば、乗除算器の正規化乗除数レジスタの先頭2ビットを加減算器に転送し、この加減算器に転送した2ビットのデータを第1の2ビット左シフト401及び選択回路402でシフトアウトして正規化乗除数レジスタ303にセットし、前記加減算器の加算回路207の結果を第2の2ビット左シフト404で2ビット左シフトし、加減算器の補数回路206の出力、加減算器の加算回路207の結果を2ビット左シフトする第2の2ビット左シフト404の出力、乗除算器の正規化乗除数レジスタ303の先頭2ビットをマージしたデータを選択し、加減算器の加算回路207の結果の最上位ビットを加減算器の桁合わせ結果レジスタ入力選択回路408に転送し、加減算器の桁合わせ結果レジスタ205を第3の1ビット左シフト407で1ビット左シフトし、加減算器の桁合わせ結果レジスタ入力選択回路408で、加減算器の桁合わせ回路203の出力、第3の1ビット左シフト407の出力、及び加算回路207の結果の最上位ビットをマージしたデータを選択し、桁合わせ結果レジスタ205の出力の補数回路409で加減算器の桁合わせ結果レジスタ205の出力を補数化し、ゼロ検出回路410で加減算器の補数結果レジスタ206と乗除算器の正規化乗除数レジスタ303がゼロであるか否かを検出し、演算制御回路105にゼロ情報を送出することにより、加減算器と乗除算器から構成される一般的な浮動小数点演算装置に、3個のシフト、補数回路、2個のデータバスという少量の回路を付加するだけで、平方根演算可能な浮動小数点演算装置を実現することができる。

【0046】また、さらに加算回路の結果の最上位ビットとオーバーフロー値を比較する誤動作検出回路を設けることにより加算回路オーバーフローの誤動作を検出することができる。

【0047】図8は、本実施例の非回復型平方根アルゴリズム処理の手順を示すフローチャートである。

【0048】図8において、 $a_{2k+1} \sim a_n$ は、図1の正規化乗除数レジスタ303に左詰めで格納され、 a_n より右のビットには0を格納する。 $q_1 \sim q_n$ は、図1の桁合わせ結果レジスタ205のビット(1, 0)～(8, 1)に右詰めで格納され、 q_1 より左のビットには0が入る。 R_k は図1の補数結果レジスタ206に右詰めで格納され、上位のビットには符号を格納する。

【0049】ステップ522の $R_{k+1} = R_k + D_k$ または512の $R_{k+1} = R_k - D_k$ の計算は、図1の加算回路207で $R_{k+1} = R_k + (\pm D_k)$ と変形しておこなう。

($\pm D_k$)は、図1の補数回路409へ桁合わせ結果レジスタ205に格納の $q_1 \sim q_k$ を入力し作成する。 $q_k = 0$ の時($\pm D_k$)は $+D_k$ となり、その値は($q_1 \dots q_{k-1}$)とする。 $q_k = 1$ の時($\pm D_k$)は $-D_k$ となり、その値は($q_1 \dots q_{k-1}$)を2の補数化した値($\neg q_1 \dots \neg q_{k-1}$)とする。

【0050】平方根の繰返し演算時、加算回路207での計算が常に真の減算となることは、図8より $q_k = 0$ の時 $R_{k+1} = R_k + D_k$ の R_k は負であり、 $q_k = 1$ の時 $R_{k+1} = R_k - D_k$ の R_k は、正であることから明らかである。真の減算であるため加減算回路207の演算結果の最上位ビットが R_{k+1} の符号を、オーバーフロー(406)は、 $R_{k+1} \geq 0$ の時 $q_{k+1} = 1$ 、 $R_{k+1} < 0$ の時 $q_{k+1} = 0$ となることによりオーバーフロー406 = q_{k+1} となる。平方根の繰返し演算時は加算回路207の結果の最上位ビットのかわりにオーバーフロー(406)を使用しても問題ない。

【0051】図8のステップ503と504による $R_k = 0$ 、 $a_{2k+1} \sim a_n = 0$ の判定は、図1のゼロ検出回路410で行なう。ゼロ検出回路410で検出されたゼロ検出信号は、演算制御回路105に報告される。演算制御回路105はゼロ検出信号により平方根の繰返し演算を中止し終了処理を行なう。

【0052】図9は、一般的な浮動小数点演算装置の仮数部の概略構成を示すブロック図であり、2は加減算器、3は乗除算器、101は第2オペランドレジスタ、102は第1オペランドレジスタ、103、201、202は選択回路、104は演算結果レジスタ、203は桁合わせ回路、204は補数回路、205は桁合わせ結果レジスタ、206は補数結果レジスタ、207は加算回路、208は加算中間結果レジスタ、209、301、302は正規化回路、303は正規化乗除数レジスタ、304は正規化被乗除数レジスタ、305は乗除算回路である。

【0053】図9に示すように、通常の加算回路の演算結果は、パリティにより誤動作の検出を行なっているが、オーバーフローなどのキャリは、誤動作の検出を行っていない。平方根演算の繰返し演算時は加算回路結果の最上位ビット=加算回路のオーバーフローが必ず成立することより、誤動作検出回路411は、加算回路の結果の最上位ビット=加算回路オーバーフローの時、誤動作報告を行なう。

【0054】また、通常の加算回路は、その結果よりもオーバーフローなどのキャリを早く求めるよう回路を設計する。そのため、加算回路207の結果の最上位ビットの符号を桁合わせ結果レジスタ入力選択回路に転送するかわりに、オーバーフロー406を桁合わせ結果レジスタ入力選択回路に転送した方が遅延時間を短くできる。以上、本発明を実施例に基づき具体的に説明したが、本発明は、前記実施例に限定されるものではなく、本発明の要旨を逸脱しない範囲において、種々変更し得ることは勿論である。

【0055】

【発明の効果】本願において開示された発明の代表的なものによって得られる効果を簡単に説明すれば、以下のとおれである。

【0056】加減算器と乗除算器から構成される一般的な浮動小数点演算装置に3個のシフタ、補数回路、2個のデータバスという少量の回路を付加することにより、平方根演算可能な浮動小数点演算装置を実現することができる。

【0057】また、加算回路の結果の最上位ビットとオーバーフロー値を比較する誤動作検出回路を設けることにより、加算回路オーバーフローの誤動作を検出することができる。

【図面の簡単な説明】

【図1】本発明による一実施例の浮動小数点演算装置の仮数部の概略構成を示すブロック図である。

【図2】本実施例の正規化乗除数レジスタ入力選択回路の動作を説明するための図である。

【図3】本実施例の補数結果レジスタ入力選択回路の動作を説明するための図である。

【図4】本実施例の桁合わせ結果レジスタ入力選択回路の動作を説明するための図である。

【図5】本実施例の桁合わせ結果レジスタ出力補数回路の動作を説明するための図である。

【図6】本実施例の浮動小数点演算装置で実行する浮動小数点平方根命令の処理手順を示すフローチャートである。

【図7】本実施例の浮動小数点演算装置の仮数部の動作を模式的に示した図である。

【図8】本実施例の非回復型平方根アルゴリズムの処理手順を示すフローチャートである。

【図9】一般的な浮動小数点演算装置の概略構成を示す

ブロック図である。

【図10】従来の回復型平方根アルゴリズムの処理手順を示すフローチャートである。

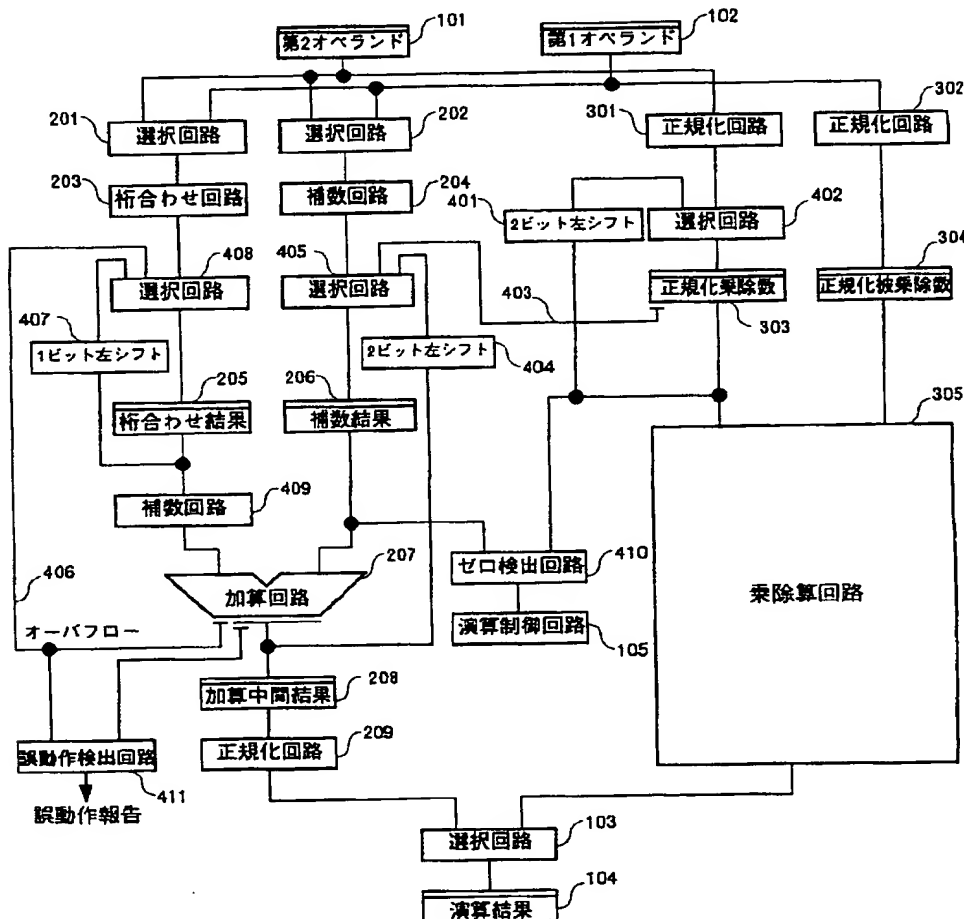
【符号の説明】

2…加減算器、3…乗除算器、101…第2オペランドレジスタ、102…第1オペランドレジスタ、103、201、202、402、405、408…選択回路、104…演算結果レジスタ、105…演算制御回路、203…桁合わせ回路、204、409…補数回路、20*

*5…桁合わせ結果レジスタ、206…補数結果レジスタ、207…加算回路、208…加算中間結果レジスタ、209、301、302…正規化回路、303…正規化乗除数レジスタ、304…正規化被乗除数レジスタ、305…乗除算回路、401、404…2ビット左シフタ、403、406…データバス、407…1ビット左シフタ、410…ゼロ検出回路、411…誤動作検出回路。

【図1】

図 1



【図2】

図 2

バイト位置	1	2	3	4	5	6	7	8			
ビット位置	0-7	0-7	0-7	0-7	0-7	0-7	0-7	0	1	2	3

選択条件

平方根演算命令以外
平方根演算命令
初期設定時

繰り返し演算時

選択データ

正規化回路出力

正規化回路出力

正規化乗除数レジスタ左2ビットシフト結果($a_{2k+1} \sim a_n$)

0 0

【図3】

図 3

バイト位置	1	2	3	4	5	6	7	8			
ビット位置	0-7	0-7	0-7	0-7	0-7	0-7	0-7	0	1	2	3

選択条件

平方根演算命令以外
平方根演算命令
初期設定時

繰り返し演算時

丸め終了処理時
途中終了処理時
($R_k=0$ & $a_{2k+1} \sim a_n=0$)

選択データ

補数回路出力

ゼロ

a_1 a_2

加算回路出力左2ビットシフト結果(R_{k-1})

a_{2k+1} a_{2k+2}

ゼロ

1 0

ゼロ

【図4】

図 4

バイト位置	1	2	3	4	5	6	7	8			
ビット位置	0-7	0-7	0-7	0-7	0-7	0-7	0-7	0	1	2	3

選択条件

平方根演算命令以外
平方根演算命令
初期設定時

繰り返し演算時
終了処理時

選択データ

桁合わせ回路出力

ゼロ

桁合わせ結果レジスタ左1ビットシフト結果($q_1 \sim q_{k-1}$)

q_k 0 0

【図5】

図 5

バイト位置	1	2	3	4	5	6	7	8			
ビット位置	0-7	0-7	0-7	0-7	0-7	0-7	0-7	0	1	2	3

選択条件

選択データ

平方根演算命令以外

桁合わせ結果レジスタ出力

平方根演算命令

繰り返し演算 $q_k=1$ 時
 \neg (桁合わせ結果レジスタ出力ビット(1,0)~(8,1) ($q_1 \sim q_k$))

1 1

繰り返し演算 $q_k=0$ 時
 (桁合わせ結果レジスタ出力ビット(1,0)~(8,1) ($q_1 \sim q_k$))

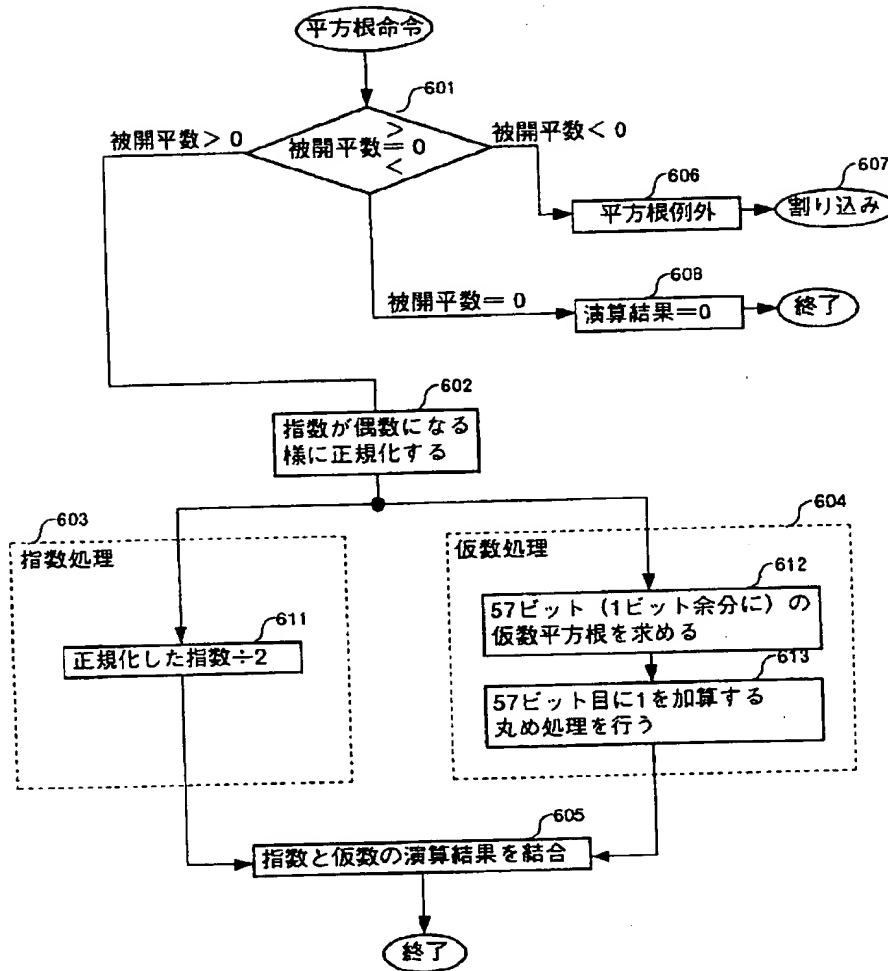
1 1

終了処理時

桁合わせ結果レジスタ出力 ($q_1 \sim q_k$ 0 0)

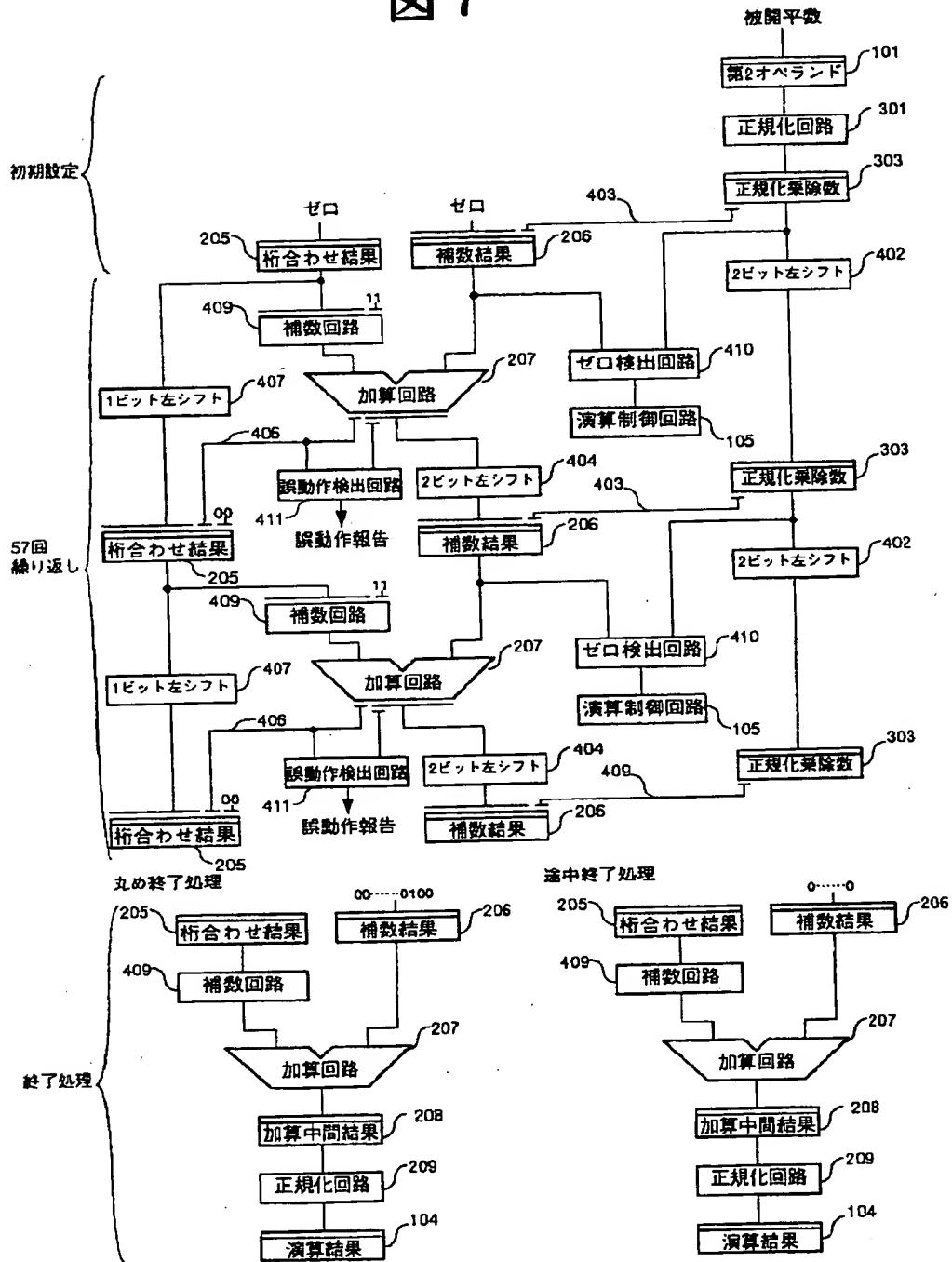
【図6】

図 6



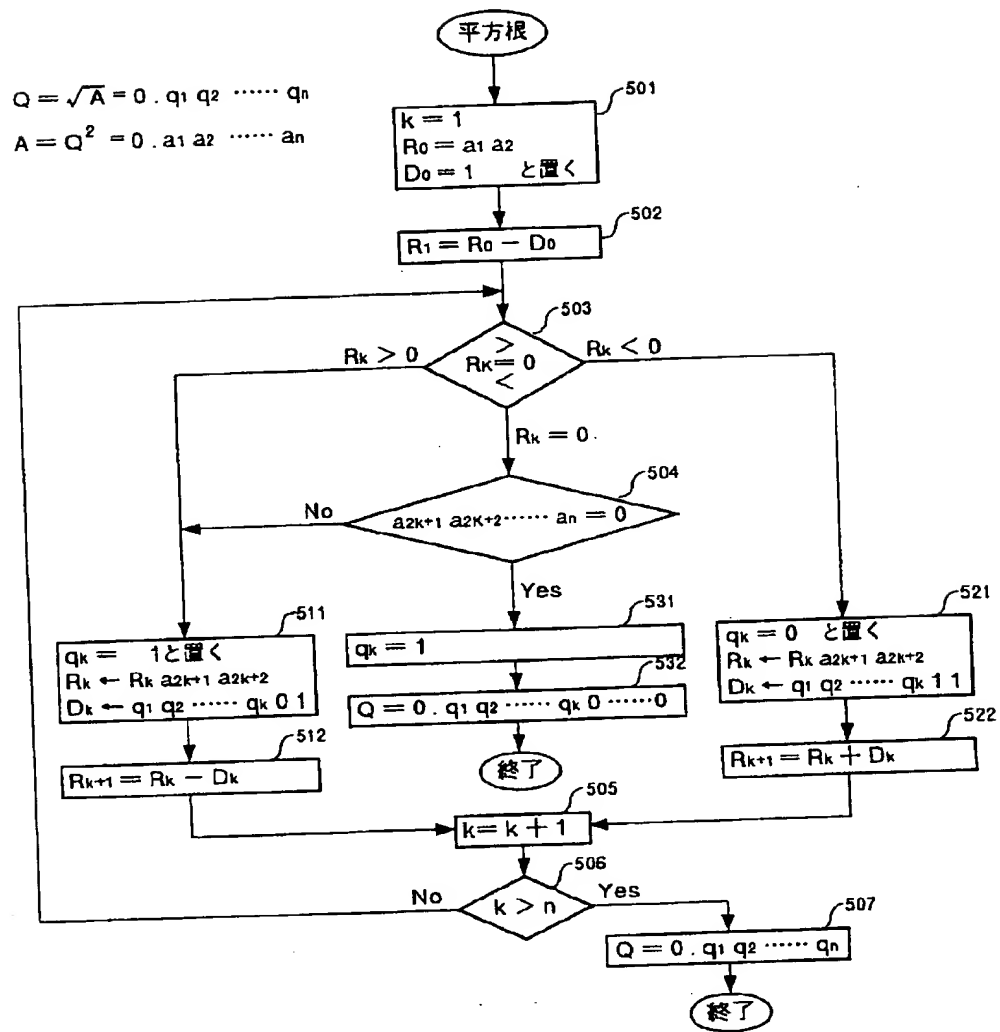
【図7】

図7



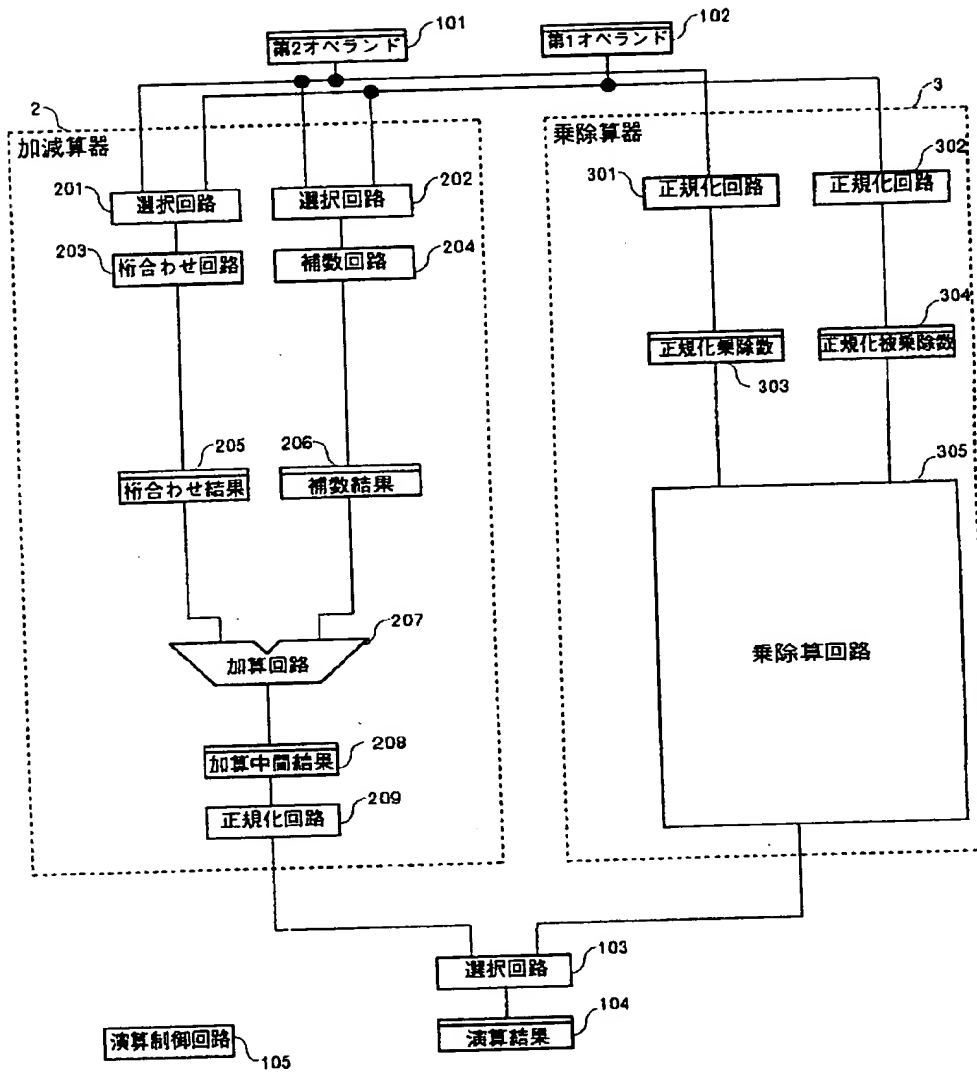
【図8】

図 8



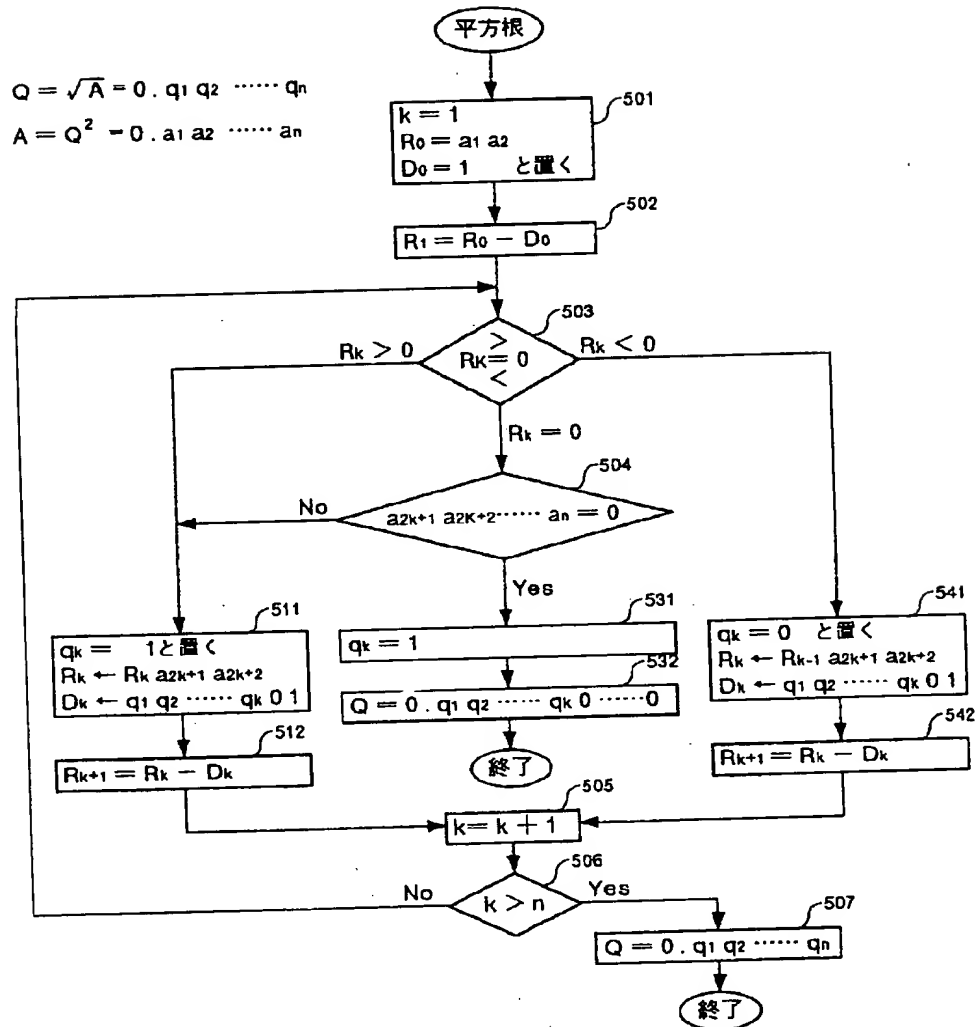
【図9】

図 9



【図10】

図 10



フロントページの続き

(72)発明者 増田 好徳
 神奈川県秦野市堀山下1番地 日立コンピ
 ュータエンジニアリング株式会社内

(72)発明者 高橋 千秋
 神奈川県秦野市堀山下1番地 日立コンピ
 ュータエンジニアリング株式会社内